IMPLEMENTASI ALGORITMA DIJKSTRA DALAM MENGANALISIS RUTE TERPENDEK DAN EFISIENSI JARAK TEMPUH DARI STASIUN KERETA API MEDAN MENUJU 6 KAMPUS DI AREA MEDAN ESTATE

Aldo Bonifasius Simbolon, Delvita Aulia Artika, Yoseph Christian Sitanggang, Putri Harliana
Ilmu Komputer, Universitas Negeri Medan
Jl. Willem Iskandar Ps. V Medan, Indonesia
aldo.4233250031@mhs.unimed.ac.id

ABSTRAK

Pesatnya perkembangan teknologi informasi memberikan solusi dalam berbagai aspek kehidupan, termasuk transportasi dan navigasi. Namun, tantangan menemukan rute perjalanan terpendek dan efisien masih kerap dialami, terutama di wilayah perkotaan seperti Medan. Penelitian ini mengimplementasikan algoritma Dijkstra untuk menganalisis rute terpendek dari Stasiun Kereta Api Medan ke enam kampus di area Medan Estate: Universitas Negeri Medan, Universitas Islam Negeri Sumatera Utara, Universitas Medan Area, Universitas Amir Hamzah, Politeknik Pariwisata Medan, dan Institut Teknologi Sawit Indonesia. Data jarak diperoleh dari Google Earth, dimodelkan sebagai graf berbobot, dan diproses menggunakan Python. Hasil penelitian menunjukkan algoritma Dijkstra secara efektif menentukan rute terpendek dengan waktu tempuh optimal. Studi ini diharapkan memberikan kontribusi dalam perencanaan perjalanan serta menjadi dasar pengembangan aplikasi navigasi berbasis lokal.

Kata kunci : *Algoritma Dijkstra, Efisiensi Jarak, Python, Analisis Rute*

1. PENDAHULUAN

Perkembangan teknologi dan juga informasi yang semakin berkembang sangat memberikan dampak yang positif bagi manusia dalam menyelesaikan berbagai tantangan di kehidupan sehari-hari, termasuk dalam bidang transportasi, logistik, dan navigasi. Namun sering kali masyarakat menemukan masalah dalam menemukan rute perjalanan yang efisien yang dapat menghemat waktu dan sumber daya terutama pada perkotaan yang memiliki lalu lintas yang macet dan mobilitas masyarakat yang tinggi sehingga sulit untuk mencari rute terpendek atau yang paling efisien untuk mencapai suatu tujuan. Masalah ini semakin rumit ketika melibatkan banyak lokasi tujuan.

Medan merupakan salah satu kota metropolitan terbesar di Indonesia yang menjadi pusat aktivitas ekonomi, pendidikan, dan sosial di wilayah Sumatera Utara. Kota ini memiliki banyak institusi pendidikan tinggi yang tersebar di berbagai lokasi strategis, di antaranya Universitas Negeri Medan (UNIMED), Universitas Islam Negeri Sumatera Utara (UINSU), Universitas Medan Area (UMA), Universitas Amir Hamzah, Politeknik Pariwisata Medan (Poltekpar), dan Institut Teknologi Sawit Indonesia (ITSI). Lokasi kampus-kampus ini menjadi tujuan harian bagi ribuan mahasiswa, dosen, dan tenaga kependidikan yang membutuhkan aksesibilitas yang mudah dan efisien. Namun, dengan infrastruktur transportasi yang terbatas dan kondisi lalu lintas yang sering macet, perjalanan ke lokasi-lokasi ini sering kali menjadi tantangan tersendiri.

Untuk mengatasi masalah ini pendekatan teori graf menjadi salah satu solusi yang meyakinkan untuk menganalisis dan menentukan rute perjalanan yang paling optimal. Salah satu algoritma yang paling banyak digunakan dalam teori graf adalah algoritma Dijkstra, yang pertama kali diperkenalkan oleh Edsger W. Dijkstra pada tahun 1956. Algoritma ini dirancang untuk menemukan jalur terpendek antara dua simpul dalam graf berbobot, dengan efisiensi tinggi dan kompleksitas yang terukur. Algoritma Dijkstra memiliki keunggulan dalam kecepatan perhitungan dan akurasi, sehingga sangat cocok untuk diterapkan dalam perencanaan rute perjalanan, sistem navigasi, dan dalam pengelolaan jaringan transportasi.

Ada beberapa penelitian yang telah dilakukan untuk menentukan rute tependek dan efisien dalm suatu perjalanan seperti yg dilakukan Bunaen [1] dimana pada penelitiannya mereka menerapkan algoritma dijkstra yang cukup populer dan mudah di implementasikan untuk menentukan rute terpendek dari pusat kota Surabaya ke tempat bersejarah, dimana pada penelitian digunakan beberapa sampel tempat bersejarah di Surabaya lalu menggunkan teknologi Google Earth untuk menggambar lintasan antar titik-titik sampel lokasi. Dengan menggunakan teori graf maka dapat ditentukan jarak terpendek antar titik dengan dimulai dari simpul dengan bobot terkecil. Dan dilakukan pengimplementasian ke dalam bahasa pemprograman C++ dengan memasukkan bobot dari setiap simpul.berdasarkan kesimpulan penelitian yang telah dilakukan didapat bahwa algortima dijkstra dapat menentukan jalur terpendek dari sebuah jalur perjalanan dengan menentukan vertex awal dan vertex tujuan serta membandingkan nilai dari masing-masing vertex. Pada penelitian lain yang dilakukan oleh Yedrizal [2] dimana pada penelitiannya Ia menggunakan algoritma heuristik untuk pendistribusian barang. Dimana permasalahan dalam pengiriman barang adalah barang harus sampai tepat waktu sampai tujuan namun karena lokasi yang akan

dituju banyak dan luas ke masing-masing daerah sangat mempersulit distributor. Metode ini dilakukan dengan mengumpulkan data berupa titik dan jarak yang dilewati saat pendistribusian lalu dikelompokkan dalam grap terhubung. Titik-titk tersebut memiliki jarak yang berbeda antar titik yang dapat digunakan untuk mencari solusi terbaik untuk rute terpendek. Kemudian dilakukan perhitungan untuk mencari kelompok titik dengan nilai paling kecil. Dalam kesimpulan penelitian ini disebutkan bahwa banyak proses perhiitungan nilai yang digunakan maka akan semakin besar peluang untuk penentukn rute arah mana yang akan dituju sehingga bisa meminimalisir waktu,jarak,dan biaya ynag dibutuhkan.

Penelitian ini bertujuan untuk mengimplementasikan algoritma Dijkstra dalam menganalisis rute terpendek dari Stasiun Kereta Api Medan ke enam kampus di Kota Medan Estate, yaitu UNIMED, UINSU, UMA, Amir Hamzah, Poltekpar, dan ITSI. Selain itu, penelitian ini juga berfokus pada evaluasi efisiensi jarak yang dapat dicapai dengan menggunakan algoritma tersebut dimana untuk implementasi dilakukan menggunakan bahasa pemrograman Python.

Dengan memanfaatkan algoritma Dijkstra, penelitian ini diharapkan mampu memberikan solusi praktis bagi masyarakat, terutama mahasiswa dan tenaga kerja, untuk merencanakan perjalanan mereka dengan lebih baik. Selain itu, temuan dalam penelitian ini juga dapat digunakan sebagai dasar untuk pengembangan aplikasi navigasi lokal yang relevan dengan kebutuhan kota-kota besar di Indonesia.

2. TINJAUAN PUSTAKA

2.1. Google Maps

Pada era sekarang untuk mengetahui rute suatu perjalanan dapat menggunakan suatu aplikasi yaitu Google maps. [3] Google Maps merupakan aplikasi peta digital yang digunakan untuk memberikan kita arahan dari lokasi kita berada menuju lokasi yang ditentukan dan terbukti dapat memberikan arahan karena banyak orang memakainya, akan tetapi Google Map belum memberikan kita kepastian apakah teknologi yang diimplementasikan untuk memberikan hasil kepada pengguna merupakan hasil terbaik dalam hal ini jalur terpendek. [1] Tidak mudah memang memilih langsung rute terpendek melalui google maps keadaan bisa saja berubah drastis saat dilihat waktu tempuhnya ternyata sebentar, rute tidak berwarna orange atau merah yang menandakan kepadatan atau kemacetan, tetapi dilokasi ternyata situasi yang berbanding terbalik terjadi kepadatan lalu lintas sehingga mengakibatkan lamanya waktu perjalanan. Permasalahan ini biasa disebut dengan TSP (Travelling Salesman Problem) [2].

2.2. Traveling Salesman Problem

Traveling Salesman Problem merupakan permasalahan klasik yang popular dalam penelitian di bidang optimasi kombinatorika Permasalahan ini merupakan permasalahan optimasi diskrit yang bertujuan untuk menentukan rute dari sesorang sales untuk mengunjungi beberapa tempat tepat satu kali dan diakhir perjalanan akan kembali ke tempat dimana perjalanan dimulai. Sehingga untuk itu diperlukan solusi permasalahan tentang analisis rute efisien [4].

2.3. Algoritma Dijkstra

Algoritma Dijkstra banyak diteliti untuk diaplikasikan pada sistem pencarian rute terpendek. Algoritma ini ditemukan oleh Edsger Dijkstra, seorang ilmuwan komputer asal Belanda. Cara kerja algoritma Dijkstra adalah dengan stategi greedy yaitu pada setiap langkahnya memilih sisi dengan nilai terkecil yang menghubungkan simpul/node yang telah terpilih dan yang belum terpilih. Prinsip algoritma Dijkstra adalah mencari dua jalur terkecil [5].

2.4. Graf

Input dari algoritma Dijkstra adalah graf berarah dan berbobot, G dan simpul sumber s pada G.V adalah himpunan dari semua simpul pada graf G. Setiap sisi graf ini merupakan sepasang simpul (u, v) yang melambangkan hubungan dari simpul u ke simpul v. [6] Graf berarah D adalah pasang berurutan dari dua himpunan V(D) dan Γ(D), yaitu himpunan berhingga tak kosong yang anggota-anggotanya disebut titik dan himpunan berhingga (boleh kosong) yang anggota-anggotanya disebut busur sedemikian hingga setiap busur merupakan pasang berurutan dari dua titik V(D. [7]. Graf merupakan suatu diagram yang memuat informasi tertentu jika diinterpretasikan secara tepat. Graf digunakan untuk menggambarkan berbagai macam struktur yang ada dalam kehidupan sehari-hari, bertujuan sebagai visualisasi objek-objek agar lebih mudah dimengerti. Representasi visual dari graf adalah dengan menyatakan objek dinyatakan sebagai noktah, bulatan, atau titik, sedangkan hubungan antara objek dinyatakan dengan garis. Secara umum graf dibagi menjadi dua, yakni graf sederhana dan graf tak sederhana. Sedangkan berdasarkan orientasi arahnya graf dibagi menjadi dua yaitu graf berarah dan graf tak berarah.

3. METODE PENELITIAN

Penelitian ini menggunakan metode pengamatan langsung yang bertujuan untuk memperoleh data yang akurat serta mendukung analisis yang komperhensif terkait penerapan algoritma Dijkstra dalam menganalisis rute terpendek dan efisiensi dari stasiun kereta api ke 6 Kampus yang berada di area Medan Estate Kab. Deli Serdang. Pengamatan langsung dilakukan dengan memanfaatkan data jarak yang diperoleh dari Google Earth, dengan pengimplementasian menggunakan aplikasi editor Visual Studio Code dengan bahasa pemprograman Python. Alur penelitian ini meliputi pengumpulan data, pemodelan graf, implementasi algoritma, pengujian menggunakan python, serta menganalisis

hasil untuk menentukan rute optimal yang sudah dihasilkan.

3.1. Pengumpulan Data

Data perhitungan operasional diambil dari Google Earth yang dinyatakan dalam meter. Google Earth adalah sebuah aplikasi pemetaan yang sangat interaktif yang diluncurkan oleh perusahaan google. Google earth sangat menarik karena menampilkan peta bola dunia, foto satelit, keadaan topografi suatu daerah, bangunan, jalan raya dan lokasi manapun di penjuru bumi [8].

Kami memilih 17 titik atau node untuk analisis lebih lanjut rute yang akan digunakan nantinya. Google Earth membantu dan memudahkan kami dalam menghitung jarak dari tiap-tiap titik yang sudah kami tentukan sebelumnya seperti pada Gambar 1

.

Gambar 1. Rute jalan dari stasiun kereta api ke 6 kampus medan estate

Berdasarkan gambar 1 dapat dilihat bahwa titik A merupakan Stasiun Kereta Api Medan, titik I merupakan Universitas Medan Area, titik L merupakan Universitas Negeri Medan, titik Q merupakan Universitas Amir Hamzah, titik M merupakan Universitas Islam Negeri Sumatera Utara, titik O merupakan Politeknik Parawisata Medan, dan titik P merupakan Institut Teknologi Sawit Indonesia. Setelah dilakukan pengukuran menggunakan Google Earth, diperoleh data rute antar titik sebagai berikut.

Tabel 1. Jarak dari setiap titik

Rute	Jarak (meter)
C-A	216
A-B	516
B-F	1,019
F-G	230
G-H	2,272
H-E	1,266
D-E	4,080
E-I	690
I-J	111
K-J	676
L-K	487
L-Q	562
Q-M	580
M-N	880
O-N	301
P-O	895

Keterangan:
A = Stasiun Kereta Api Medan
B = Simpang Jl. M.T. Haryono
C = Simpang Jl. Pulau Pinang
D = Simpang Jl. Perintis Kemerdekaan
E = Jl. Kolam
F = Simpang M.H. Thamrin
G = Jl. Wahidin
H = Simpang Jl. Mandala Bypass
I = Universitas Medan Area
J = Kantin Kampus UMA
K = Jl. Selamat Ketaren
L = Universitas Negeri Medan
M = Universitas Islam Negeri Indonesia
N = Simpang Jl. Rumah Sakit H.
O = Politeknik Parawisata Indonesia
P = Institut Teknologi Sawit Indonesia
Q = Universitas Amir Hamzah

3.2. Pemodelan Graf

Algoritma dijkstra ini digunakan untuk menemukan rute terpendek dari satu titik awal ke titik lainnya dalam sebuah graf berbobot positif. Dalam algoritma dijkstra sebuah graf direpresentasikan sebagai kumpulan simpul (nodes) dan sisi(edges) yang menghubungkan simpul-simpul tersebut. Setiap sisi memiiki bobot(weight) yang mewakili biaya atau jarak antara dua simpul.

Representasi graf pada algoritma dijkstra dilakukan dengan melihat daftar ketetanggaan yang menyimpan setiap simpul beserta simpul-simpul tetangga yang berhubungan langsung dan bobotnya. Berdasarkan data yang telah kami kumpul kan pada tabel 1, berikut adalah daftar ketetanggaan (Adjacency List) dari setiap simpul :

Tabel 2. Daftar ketetanggaan dari setiap simpul

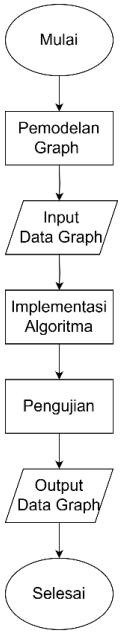
Simpul	Tetangga (simpul : jarak (meter))
A	B : 516, C : 216
B	F : 1,019
C	-
D	E : 4,080
E	I : 690
F	G : 230
G	H : 2,272
H	-
I	J : 111
J	K : 676
K	L : 487
L	Q : 562
M	N : 880, Q : 580
N	O : 301
O	P : 895
P	-
Q	M : 580

3.3. Pembentukan *Flowchart*

Flowchart adalah bentuk diagram yang menggambarkan algoritma atau langkah-langkah berurutan dari instruksi dalam sebuah sistem. Gambaran umum *flowchart* dapat membantu

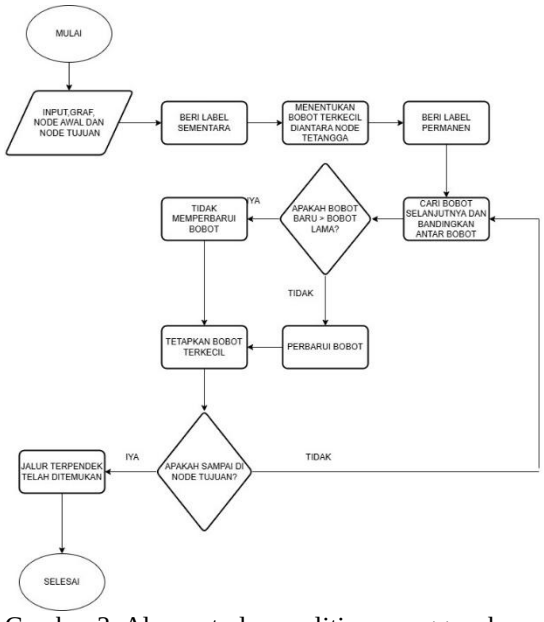
menyelesaikan masalah yang mungkin muncul selama pengembangan sistem. Pada dasarnya, *flowchart* digambarkan dengan simbol-simbol. Setiap simbol menunjukkan langkah tertentu. Namun, untuk menghubungkan satu operasi ke operasi lainnya, garis penghubung digunakan [9].

Gambar 2 merupakan *flowchart* dari implementasi algoritma Djikstra yang akan diterapkan menggunakan Python.



Gambar 2. Alur metode penelitian menggunakan python

Sementara itu, gambar 3 merupakan *flowchart* dari implementasi algoritma Djikstra yang akan dihitung secara manual.



Gambar 3. Alur metode penelitian menggunakan perhitungan manual

3.4. Implementasi Algoritma Dijkstra

Pengimplementsian algoritma dijkstra untuk menemukan jalur terpendek dari titik awal ke semua titik lain dilakukan dengan menggunakan rumus dijkstra. Rumus dijkstra bekerja dengan cara memperbarui jarak terpendek sementara dari titik awal (A) ke titik lainnya, yang di ulang dengan iterasi. Dengan rumus dijkstra nya adalah:

distance(v)=min(distance(v),distance(u)+weight(u,v))

dengan :

- distance(v) adalah jarak saat ini ke simpul v,
- distance(u) adalah jarak ke simpul u,
- weight(u,v) adalah bobot (jarak) sisi dari u ke v.

Dimana pengimplementasiaannya dimulai dengan:

3.4.1. Inisialisasi

Tentukan semua simpul yang ada dalam graf yaitu:A,B,C,D,E,F,G,H,I,,K,L,M,N,O,P,Q. kemudian set jarak dari titik awal (titik A) ke lain. Semua titik lain buat sebagai tak terhingga, kecuali titik A ke titik A adalah 0.

3.4.2. Iterasi

Iterasi 1:

Dimulai dari simpul dengan jarak terpendek yang belum dikunjungi (dimulai dari A). Dari simpul A ada 2 tetangga yaitu : B dan C . Dengan jarak

A-B : min(∞,0 + 516)= 516

A-C : min(∞,0 + 216)= 216

Sehingga simpul yang telah dikunjungi setelah iterasi 1:A,C,B

Iterasi 2:

Pilih simpul berikutnya yang belum dikunjungi (C). Namun simpul C tidak memilih tetangga yang bisa diperbarui, sehingga dilanjutkan ke simpul terpendek berikutnya yaitu B(516) dengan simpul tetangga F(1,019). Dengan jarak :

B-F: min(∞,516+1,019) = 1,535

Sehingga simpul yang telah dikunjungi setelah iterasi 2:A,C,B,F

Iterasi 3:

Pilih simpul dengan jarak terpendek yang belum di kunjungi yaitu F(1,535) dengan tetangga simpul tersebut adalah G(230). Dengan jarak :

F-G : min(∞,1,535+230)=1,765.

Sehingga simpul yang telah dikunjungi setelah iterasi 3:A,C,B,F,G

Iterasi 4:

Pilih simpul dengan jarak terpendek yang belum di kunjungi yaitu G(1,765) dengan tetangga simpul tersebut adalah H(2,272). Dengan jarak :

G-H : min(∞,1,765+2,272)=4,037.

Sehingga daftar jarak setelah iterasi 4 yaitu:

Sehingga simpul yang telah dikunjungi setelah iterasi 5:A,C,B,F,G,H.

Iterasi 5:

Pilih simpul dengan jarak terpendek yang belum di kunjungi yaitu H(4,037) dengan tetangga simpul tersebut adalah E(1,266). Dengan jarak :
G-H : min(∞,4,037+1,266)=5,303
Sehingga daftar jarak setelah iterasi 5 yaitu:
Sehingga simpul yang telah dikunjungi setelah iterasi 4:A,C,B,F,G,H.E

Iterasi 6

Pilih simpul dengan jarak terpendek yang belum di kunjungi yaitu E(5,303) dengan tetangga simpul tersebut adalah I(690) . Dengan jarak:
I-J: min(∞,5,303+690)=5,993
Sehingga simpul yang telah dikunjungi setelah iterasi 6:A,C,B,F,G,H,E,I,

Iterasi 7:

Pilih simpul dengan jarak terpendek yang belum di kunjungi yaitu I(5,993) dengan tetangga simpul tersebut adalah J(111).
I-J : min(∞,5,993+111)=6,104
Sehingga simpul yang telah dikunjungi setelah iterasi 7:A,C,B,F,G,H,E,I,J,

Iterasi 8:

Pilih simpul dengan jarak terpendek yang belum di kunjungi yaitu J(6,104) dengan tetangga simpul tersebut adalah K(676) Dengan jarak:
J-K: min(∞,6,104+676)=6,780
Sehingga simpul yang telah dikunjungi setelah iterasi 8:A,C,B,F,G,H,E,I,J,K.

Iterasi 9:

Pilih simpul dengan jarak terpendek yang belum di kunjungi yaitu K(6,780) dengan tetangga simpul tersebut adalah L(487). Dengan jarak:
K-L: min(∞,6,780+487)=7,267
Sehingga simpul yang telah dikunjungi setelah iterasi 9:A,C,B,F,G,H,E,I,J,K,L.

Iterasi 10:

Pilih simpul dengan jarak terpendek yang belum di kunjungi yaitu L(7,267) dengan tetangga simpul tersebut adalah Q(562). Dengan jarak
L-Q : min(∞,7,267+562)=7,829
Sehingga simpul yang telah dikunjungi setelah iterasi 10:A,C,B,F,G,H,E,I,J,K,L,Q.

Iterasi 11:

Pilih simpul dengan jarak terpendek yang belum di kunjungi yaitu Q(7,829) dengan tetangga simpul tersebut adalah M(580). Dengan jarak:
Q-M: min(∞,7,829+580)=8,409
Sehingga simpul yang telah dikunjungi setelah iterasi 11:A,C,B,F,G,H,E,I,J,K,L,Q,M.

Iterasi 12:

Pilih simpul dengan jarak terpendek yang belum di kunjungi yaitu M(8,409) dengan tetangga simpul tersebut adalah N(880). Dengan jarak :
M-N : min(∞,8,409+880)=9,289
Sehingga simpul yang telah dikunjungi setelah iterasi 12:A,C,B,F,G,H,E,I,J,K,L,Q,M,N.

Iterasi 13:

Pilih simpul dengan jarak terpendek yang belum di kunjungi yaitu N(9,289) dengan tetangga simpul tersebut adalah O(301). Dengan jarak :
N-O: min(∞,9,289+301)=9,590
Sehingga simpul yang telah dikunjungi setelah iterasi 13:A,C,B,F,G,H,E,I,J,K,L,Q,M,N,O.

Iterasi 14:

Pilih simpul dengan jarak terpendek yang belum di kunjungi yaitu O(9,590) dengan tetangga simpul tersebut adalah P(895). Dengan jarak :
O-P : min(∞,9,590+895)=10,485
Sehingga simpul yang telah dikunjungi setelah iterasi 14:A,C,B,F,G,H,E,I,J,K,L,Q,M,N.O,P.

Sehingga berdasarkan iterasi yang dilakukan dengan algoritma dijkstra total jarak yang ditempuh adalah:10,485 meter
Dengan rute terpendeknya : A → B → F → G → H → E → I → J → K → L → Q → M → N → O → P

4. HASIL DAN PEMBAHASAN

Pengujian dan pembahasan dilakukan pada aplikasi editor Visual Studio Code dengan menggunakan bahasa pemrograman Python. Python adalah bahasa pemrograman tingkat tinggi yang pertama kali dirilis pada tahun 1991. Bahasa ini dikembangkan oleh Guido van Rossum pada tahun 1989 sebagai lanjutan dari bahasa pemrograman ABC [10].

Graf yang sudah diperoleh pada pengukuran jarak di Google Earth sebelumnya diterjemahkan kedalam bahasa pemprograman Python.

Setelah input data graf selesai dilakukan, Algoritma Dijkstra diprogram agar dapat melakukan perhitungan graf pada permasalahan Traveling Salesman Probem yang akan diselesaikan.

Kode program fungsi untuk mencari rute terpendek melalui beberapa tujuan bekerja dengan cara mencoba semua kemungkinan urutan kunjungan menggunakan fungsi *itertools.permutations.* Untuk setiap urutan, fungsi menghitung jarak antar simpul menggunakan algoritma Dijkstra. Fungsi menambahkan jarak dan jalur setiap segmen perjalanan hingga semua tujuan tercapai. Hasil berupa total jarak dan jalur terpendek disimpan jika lebih optimal dibandingkan sebelumnya. Setelah semua permutasi selesai diperiksa, fungsi mengembalikan jalur dengan jarak paling kecil.

Program ini dimulai dengan menentukan titik awal (A) dan daftar tujuan yang akan dilalui ('Q', '`O',

'L', 'I', 'M', 'P'). Menggunakan fungsi find_shortest_route, program mencari rute dengan jarak terpendek melalui semua tujuan dengan menghitung semua kemungkinan jalur menggunakan algoritma Dijkstra untuk setiap segmen perjalanan. Setelah jalur terpendek ditemukan, hasil berupa total jarak dan urutan jalur ditampilkan di dalam terminal.

Setelah semua fungsi dan kode program dimasukkan, hasil dari program tersebut didapatkan. Hasilnya, Algoritma Dijkstra berhasil menemukan jarak dan jalur terpendek yang perlu dilalui.

Berdasarkan output yang dihasilkan dari IDE Visual Studio Code tersebut, kode program menunjukkan hasil bahwa jalur terpendek untuk melewati 6 kampus dengan stasiun kereta api Medan sebagai titik awal yaitu A → B → F → G → H → E → I → J → K → L → Q → M → N → O → P, atau dalam data real, rutenya yaitu Stasiun Kereta Api Medan → Simpang Jl. M.T. Haryono → Simpang M.H. Thamrin → Jl. Wahidin → Simpang Jl. Mandala Bypass → Jl. Kolam → Universitas Medan Area → Kantin Kampus UMA → Jl. Selamat Ketaren → Universitas Negeri Medan → Universitas Amir Hamzah → Universitas Islam Negeri Indonesia → Simpang Jl. Rumah Sakit H. → Politeknik Parawisata Indonesia → Institut Teknologi Sawit Indonesia. Jarak tempuh yang perlu dilalui melalui rute tersebut yaitu 10,485 meter.

Gambar 4. Rute perjalanan dari stasiun kereta api medan menuju 6 kampus setelah perhitungan

Penelitian ini berhasil mengimplementasikan algoritma Dijkstra dalam mencari rute terpendek dan efisiensi jarak tempuh dari stasiun kereta api Medan menuju 6 kampus yang ada di area Medan Estate.

5. KESIMPULAN DAN SARAN

Dari penelitian yang telah dilakukan dapat diambil kesimpulan bahwa penggunaan algoritma dijkstrak terbukti efisien untuk menentukan rute terpendek dan jarak tempuh yang paling efisien dari stasiun kereta api Medan ke enam kampus yang berada Medan Estate dengan pengimplementasian di python yang dapat menampilkan jarak dan jalur mana yang terpendek. Dari hasil analisis pula penggunaan algoritma ini membantu dalam meminimalkan biaya dan waktu tempuh dengan mengidentifikasi rute dengan jarak terpendek dan waktu yang paling efisien yang dilkukan dengan mempertimbangkan bobot (*distance*) dari jalan yang tersedia. Walaupun penggunaan algoritma dijkstra pada python ini dinilai efektif dan efisien namun ada nya ketidaksesuaian kondisi data real-time nya seperti kondisi jalan dan jadwal kereta api dapat menyebakan hasil yang diberikan kurang relevan selain itu adanya perubahan infrastruktur jalan dan pola lalu lintas yang berubah dapat menyebabkan hasil yang di terima kurang sesuai. Meskipun penelitian pengimplementasian Algoritma Dijkstra ini sudah sesuai dengan yang diharapkan, penelitian ini masih dapat dikembangkan lebih jauh lagi dengan menggunakan metode-metode seperti mengintegrasikan data lalu lintas real-time dengan memanfaatkan API dari layanan navigasi seperti Google Maps, sehingga hasil analisis rute dapat lebih relevan dengan kondisi jalan saat itu. Selain itu, perbandingan dengan algoritma lain seperti A* atau Bellman-Ford dapat dilakukan untuk mengevaluasi efisiensi dan kecepatan algoritma dalam berbagai skenario. Penelitian juga dapat memperluas fokus dengan menganalisis biaya perjalanan atau konsumsi energi, sehingga memberikan manfaat yang lebih praktis. Penggunaan pendekatan multi-criteria decision making dapat menjadi pilihan, dengan mempertimbangkan faktor tambahan seperti tingkat kemacetan, risiko keamanan, atau waktu keberangkatan. Pengimplementasian metode-metode ini dapat membuat data yang diperoleh semakin akurat dan lebih mudah untuk digunakan kedepannya.

DAFTAR PUSTAKA

[1] M. C. Bunaen, H. Pratiwi, and Y. F. Riti, "Penerapan algoritma dijkstra untuk menentukan rute terpendek dari pusat kota Surabaya ke tempat bersejarah," *J. Teknol. Sist. Inf. Bisnis-JTEKSIS*, vol. 4, no. 1, pp. 213–223, 2022.

[2] Yendrizal, "Distribusi Produk Menggunakan Metode Travelling Salesman Problem (TSP) Dengan Konsep Algoritma Heuristik," *KESATRIA: Jurnal Penerapan Sistem Informasi (Komputer & Manajemen)*, vol. 5, no. 3, pp. 818-824, 2024.

[3] R. Kusnanto and K. Handoko, "Penentuan jarak terdekat wisata kuliner menggunakan algoritma Dijkstra," *Comasiejournal*, vol. 4, no. 5, pp. 111–118, Jan. 2021.

[4] R. Yuliantari and L. Musabbikhah, "Review artikel: Analisis penggunaan algoritma Dijkstra untuk mencari rute terpendek di rumah sakit," *Edu Elektrika J.*, vol. 11, no. 1, pp. 1–5, 2022.

[5] Fatkharrofiqi and W. Gata, "Implementasi algoritma Dijkstra dalam penentuan rute terdekat menuju masjid di Perumahan Bona Indah Lebak Bulus," *J. Inf. Syst. Appl. Manag. Account. Res.*, vol. 6, no. 1, pp. 87–92, 2022.

[6] Muklason and I. G. A. Premananda, "Optimasi rute rencana perjalanan pesawat menggunakan

algoritma Late Acceptance Hill Climbing (studi kasus: Travelling Salesman Challenge 2.0)," *J. Teknol. Inf. Ilmu Komput.*, vol. 10, no. 4, pp. 893–898, 2024.

[7] Rafi'Addani, T. Turmudi, and I. Sujarwo, "Penerapan graf berarah dan berbobot untuk mengetahui influencer yang paling berpengaruh dalam penyebaran informasi pada Twitter," *J. Riset Mahasiswa Mat.*, vol. 2, no. 5, pp. 186–194, 2023.

[8] M. K. Ali, A. L. Kamal, D. Safitri, and S. Sujarwo, "Penggunaan Google Earth dalam pembelajaran IPS," *J. Teknol. Pendidikan*, vol. 1, no. 4, pp. 1–9, 2024. [Online]. Available: https://edu.pubmedia.id/index.php/JTP. DOI: https://doi.org/10.47134/JTP.V1I4.379

[9] R. Rosaly and A. Prasetyo, "Flowchart beserta fungsi dan simbol-simbol," *J. Chem. Inf. Model.*, vol. 2, no. 3, pp. 5–7, 2020.

[10] R. Suherman and N. R. Kurnianda, "Optimization of VSAT IP installation routes using the Dijkstra algorithm," *IJCT J.*, vol. 8, no. 1, pp. 153–157, 2021. [Online]. Available: http://www.ijctjournal.org/volume8/issue1/ijctv8i1p16.pdf